

OCTOBER 26, 2023

---

# AVEVA Adapters for MQTT and AVEVA Edge Data Store IIoT configuration workshop

Presented by: Ashish Jain, Evan Greavu

**AVEVA**

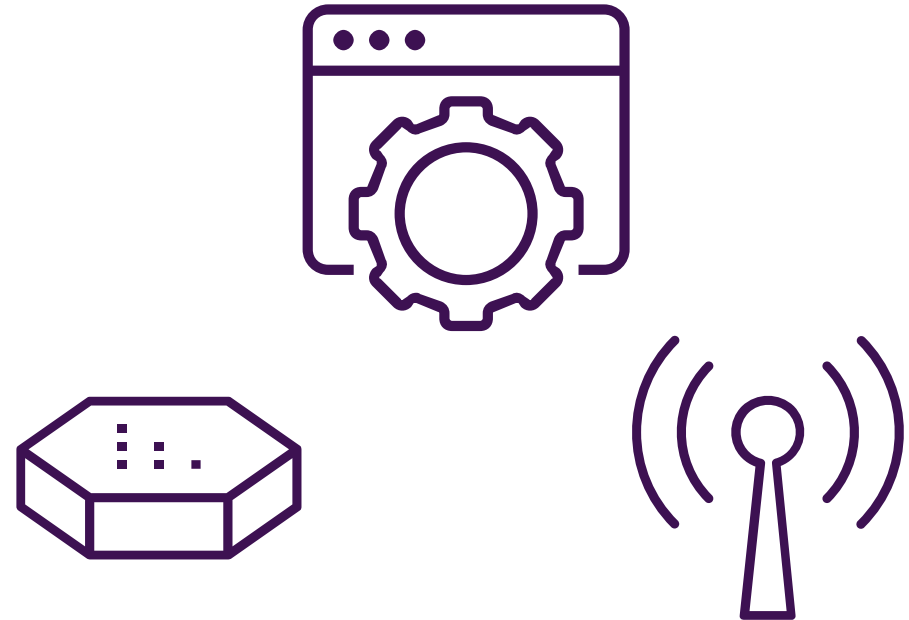
---

# Goals and takeaways

- Install and configure AVEVA Adapter for MQTT for data collection
- Install and configure AVEVA Edge Data Store for data ingress from the AVEVA Adapter
- Install both components on one Linux server
- Install EdgeCmd Utility and configure AVEVA Adapter to send data to AVEVA Edge Data Store
- Easy to setup and configure
- View the data that is set up for data collection

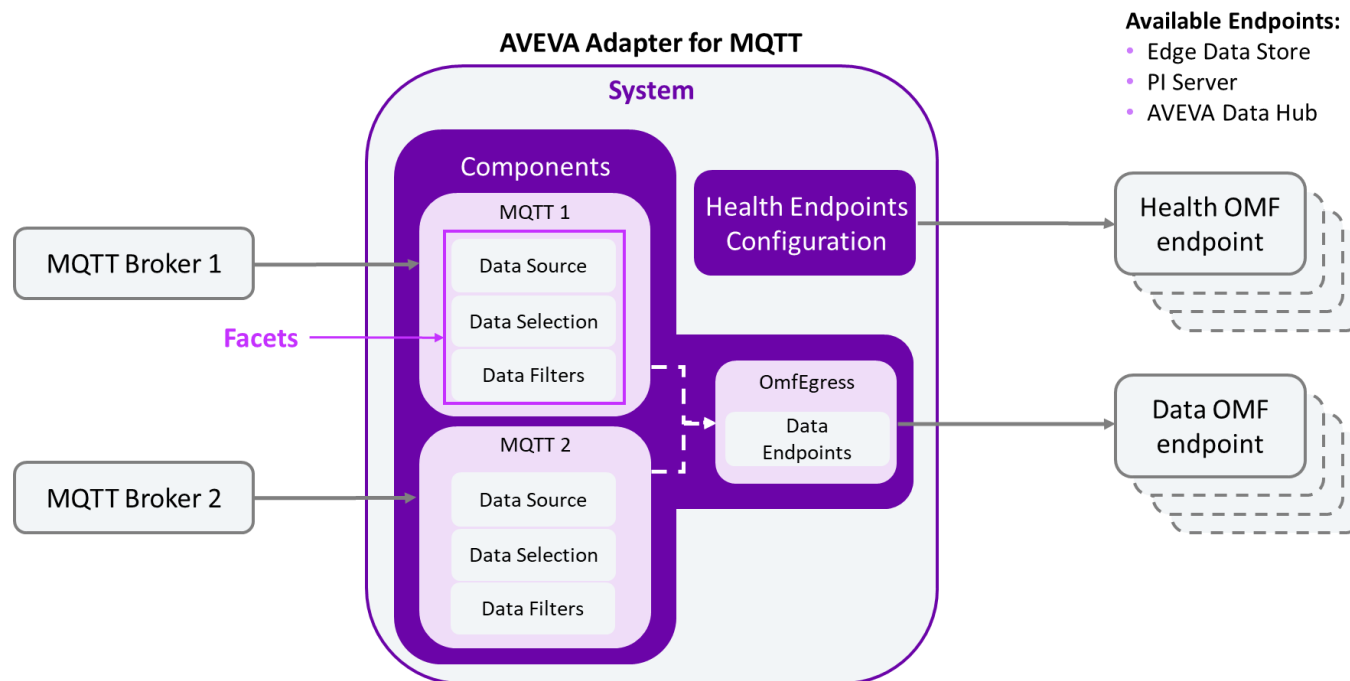
# What is EdgeCmd?

- EdgeCmd is a command line application to configure and/or administer the AVEVA Edge Data Store and the suite of AVEVA Adapters
- EdgeCmd queries are translated to HTTP queries against REST API
- Separate install kit
- Supported on Linux or Windows OS
- Easy to use with facets
- Can also use cURL and/or Postman



# What are facets?

- Sections of a configuration to make data collection unique for each adapter component
- Used with a list of operations (help, get, set, edit, add, remove, etc.)
- User-friendly names to help configure adapters easier
- Analogous to the different tabs we find in PI ICU



# Different facets available to use

Configurable facets
Components
Logging
Buffering
HealthEndpoints
DataSource
RedundantServers
ClientFailover
DataEndpoints
DataFilters
DataSelection

Read-only facets
Version
General
Diagnostics
Application
FailoverState
ClientSettings

# Pre-requisite

- The following components will be installed:
  - Edge Data Store on port: 5590
  - AVEVA Adapter for MQTT on port 5591
  - EdgeCmd utility
  - MQTT data source on port 1883
- What we will be doing:
  - Configure the adapter to connect to data source and discover for data
  - Configuration will be done using EdgeCmd
  - Use AVEVA Edge Data Store to view the results



---

# Getting started: Installing the AVEVA software components





File Edit View Search Terminal Help

ashish@AE-Linux6:~/Desktop/Installers\$

I





# Getting started: Installing the AVEVA software components

**Installing on Linux:** `sudo apt install <filename>`

**To verify:** `edgcmd get system`

```
ashish@AE-Linux5:~/Desktop/Installers$ edgcmd get System
{
  "Logging": {
    "logLevel": "Information",
    "logFileSizeLimitBytes": 34636833,
    "logFileCountLimit": 31
  },
  "HealthEndpoints": [],
  "Components": [
    {
      "componentId": "OpcUa1",
      "componentType": "OpcUa"
    },
    {
      "componentId": "Modbus1",
      "componentType": "Modbus"
    },
    {
      "componentId": "Storage",
      "componentType": "Storage"
    }
  ],
  "Buffering": {
    "bufferLocation": "/usr/share/OSIsoft/EdgeDataStore/Buffers",
    "maxBufferSizeMB": 1024,
    "enablePersistentBuffering": true
  },
  "General": {
    "enableDiagnostics": true,
    "metadataLevel": "Medium",
    "healthPrefix": null
  }
}
```

---

# Basics steps to configure an AVEVA Adapter

1. Create an AVEVA adapter component
2. Configure a data source
3. Configure egress to the data endpoint (AVEVA™PI System™, AVEVA Edge Data Store, and AVEVA™ Data Hub)
4. (optional) Configure health endpoints (AVEVA PI System, AVEVA Edge Data Store, and AVEVA Data Hub)
5. (optional) Configure data filtering
6. (optional) Discover data items
7. Configure data selection
8. Confirm data flow

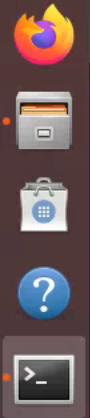
---

# Configuring the AVEVA Adapter

Create an adapter component



ashish@AE-Linux6:~/Desktop/Installers\$



# Configuring the AVEVA Adapter

## Create an AVEVA Adapter component

**Creating a component:** `edgcmd add components -type MQTTSparkplugB -id Sparkplug1 -port 5591`

**To verify:** `edgcmd get components -port 5591`

```
ashish@AE-Linux6:~/Desktop/Installers$ edgcmd get components -port 5591
[
  {
    "componentId": "0mfEgress",
    "componentType": "0mfEgress"
  },
  {
    "componentId": "Sparkplug1",
    "componentType": "MQTTSparkplugB"
  }
]
```

---

# Configuring the AVEVA Adapter

Configure a data source





File Edit View Search Terminal Help

ashish@AE-Linux6:~/Desktop/Installers\$

I



# Configuring the AVEVA Adapter

## Configure a data source

**Configure datasource:** `edgcmd set datasource -cid Sparkplug1 -file DataSource.json -port 5591`

**To verify:** `edgcmd get datasource -cid Sparkplug1 -port 5591`

```
ashish@AE-Linux6:~/Desktop/Installers$ edgcmd get datasource -cid Sparkplug1 -port 5591
{
  "hostNameOrIpAddress": "10.4.209.213",
  "port": 1883,
  "primaryHostId": null,
  "protocol": "Tcp",
  "tls": "None",
  "userName": "adapter",
  "password": "{{Sparkplug1.DataSource.Password}}",
  "clientId": "55827c94-a934-4c44-8f47-735628b231f7",
  "clientCertificateThumbprint": null,
  "clientCertificatePassword": null,
  "mqttVersion": "3.1.1",
  "validateServerCertificate": true,
  "streamIdPrefix": null,
  "defaultStreamIdPattern": "{BaseTopic}.{MetricName}"
}
```



---

# Configuring the AVEVA Adapter

Configure egress to a data endpoint



ashish@AE-Linux6:~/Desktop/Installers\$



# Configuring the AVEVA Adapter

## Configure egress to data endpoint

**Configure egress endpoint:** `edgcmd set dataEndpoints -file EgressEndpoint.json -port 5591`

**To verify:** `edgcmd get dataendpoints -port 5591`

```
ashish@AE-Linux6:~/Desktop/Installers$ edgcmd get dataendpoints -port 5591
[
  {
    "id": "EDS",
    "endpoint": "http://localhost:5590/api/v1/tenants/default/namespaces/default/omf",
    "userName": "",
    "password": "",
    "clientId": null,
    "clientSecret": null,
    "debugExpiration": null,
    "tokenEndpoint": null,
    "validateEndpointCertificate": true
  }
]
```

---

# Configuring the AVEVA Adapter

(optional) Configure health endpoints





ashish@AE-Linux6: ~/Desktop/Installers\$



# Configuring the AVEVA Adapter

(optional) Configure health endpoints

**Configure health endpoint:** `edgcmd set healthendpoints -file EgressEndpoint.json -port 5591`

**To verify:** `edgcmd get healthendpoints -port 5591`

```
ashish@AE-Linux6:~/Desktop/Installers$ edgcmd get healthendpoints -port 5591
[
  {
    "id": "EDS",
    "endpoint": "http://localhost:5590/api/v1/tenants/default/namespaces/default/omf",
    "userName": "",
    "password": "",
    "clientId": null,
    "clientSecret": null,
    "debugExpiration": null,
    "tokenEndpoint": null,
    "validateEndpointCertificate": true
  }
]
```

---

# Configuring the AVEVA Adapter

(optional) Configure data filters





File Edit View Search Terminal Help

ashish@AE-Linux6: ~/Desktop/Installers\$

I





# Configuring the AVEVA Adapter

(optional) Configure data filtering

**Apply data filter:** `edgcmd set datafilters -cid Sparkplug1 -port 5591 -file DataFilters.json`

**To verify:** `edgcmd get datafilters -cid Sparkplug1 -port 5591`

```
ashish@AE-Linux6:~/Desktop/Installers$ edgcmd get datafilters -cid Sparkplug1 -port 5591
[
  {
    "id": "duplicate",
    "absoluteDeadband": null,
    "percentChange": 1,
    "expirationPeriod": "0:01:00"
  }
]
```

---

# Configuring the AVEVA Adapter

(optional) Discover data items



ashish@AE-Linux6:~/Desktop/Installers\$



I

# Configuring the AVEVA Adapter

(optional) Discover data items

**To discover data items:** `edgcmd add discoveries -cid Sparkplug1 -id Discovery1 -port 5591`

**To verify:** `edgcmd get discoveries -cid Sparkplug1 -id Discovery1 -port 5591`

```
ashish@AE-Linux6:~/Desktop/Installers$ edgcmd get discoveries -cid Sparkplug1 -id Discovery1 -port 5591
{
  "id": "Discovery1",
  "query": null,
  "startTime": "2023-09-15T07:22:40.7544385-07:00",
  "endTime": "2023-09-15T07:23:40.9011998-07:00",
  "progress": 1,
  "itemsFound": 15,
  "newItems": 15,
  "resultUri": "http://127.0.0.1:5591/api/v1/Configuration/Sparkplug1/Discoveries/Discovery1/result",
  "autoSelect": false,
  "status": "Complete",
  "errors": null
}
```

---

# Configuring the AVEVA Adapter

Configure data selection





# Configuring the AVEVA Adapter

## Configure data selection

### Using discovery for data selection:

```
edgcmd add dataselection -cid Sparkplug1 -unselect  
-query discoveryid=Discovery1 -port 5591
```

### Outputting discovery results to file:

```
edgcmd get dataselection -cid Sparkplug1 -port 5591  
> MQTTDataSelection.json
```

### To apply data selection contents:

```
edgcmd set dataselection -cid Sparkplug1 -port 5591  
-file MQTTDataSelection.json
```

### To verify:

```
edgcmd get dataselection -cid Sparkplug1 -port 5591
```

```
ashish@AE-Linux6:~/Desktop/Installers$ edgcmd get dataselection -cid Sparkplug1 -port 5591  
[  
  {  
    "topic": "spBv1.0/My MQTT Group/NDATA/Edge Node fcfb93",  
    "metricName": "RandomShort1",  
    "selected": true,  
    "name": null,  
    "streamId": "spBv1.0/My MQTT Group/Edge Node fcfb93.RandomShort1",  
    "dataFilterId": null  
  },  
  {  
    "topic": "spBv1.0/My MQTT Group/NDATA/Edge Node fcfb93",  
    "metricName": "RandomDouble2",  
    "selected": true,  
    "name": null,  
    "streamId": "spBv1.0/My MQTT Group/Edge Node fcfb93.RandomDouble2",  
    "dataFilterId": null  
  },  
  {  
    "topic": "spBv1.0/My MQTT Group/NDATA/Edge Node fcfb93",  
    "metricName": "RandomInteger2",  
    "selected": true,  
    "name": null,  
    "streamId": "spBv1.0/My MQTT Group/Edge Node fcfb93.RandomInteger2",  
    "dataFilterId": null  
  },  
]
```

---

# Configuring the AVEVA Adapter

Confirm data flow







# Configuring the AVEVA Adapter

## Confirm data flow

Open the browser, go to <http://localhost:<EDSPort>/api/v1/tenants/default/namespaces/default/streams/> for all the streams available

Replacing the <StreamId> with a tag name to see data:

<http://localhost:<EDSPort>/api/v1/tenants/default/namespaces/default/streams/<StreamID>/Data/Last>

- Can use Grafana, 3<sup>rd</sup> party tool to visualize the data locally before egress
- Can egress the data to AVEVA PI Server or AVEVA Data Hub to use with their respective suite of software (AVEVA™ PI Vision™ for AVEVA PI Server and trend tool for AVEVA Data Hub)

---

# Configuring the AVEVA Adapter using Postman





POST http://localhost:5590/ap + ⋮



http://localhost:5590/api/v1/configuration/system/components

Save



POST ▾

http://localhost:5590/api/v1/configuration/system/components

Send ▾

Params Authorization Headers (8) **Body** Pre-request Script Tests Settings

Cookies

 none  form-data  x-www-form-urlencoded  raw  binary **JSON** ▾

Beautify

```
1 {
2   "componentId": "Sparkplug1",
3   "componentType": "MQTTSparkplugB"
4 }
```

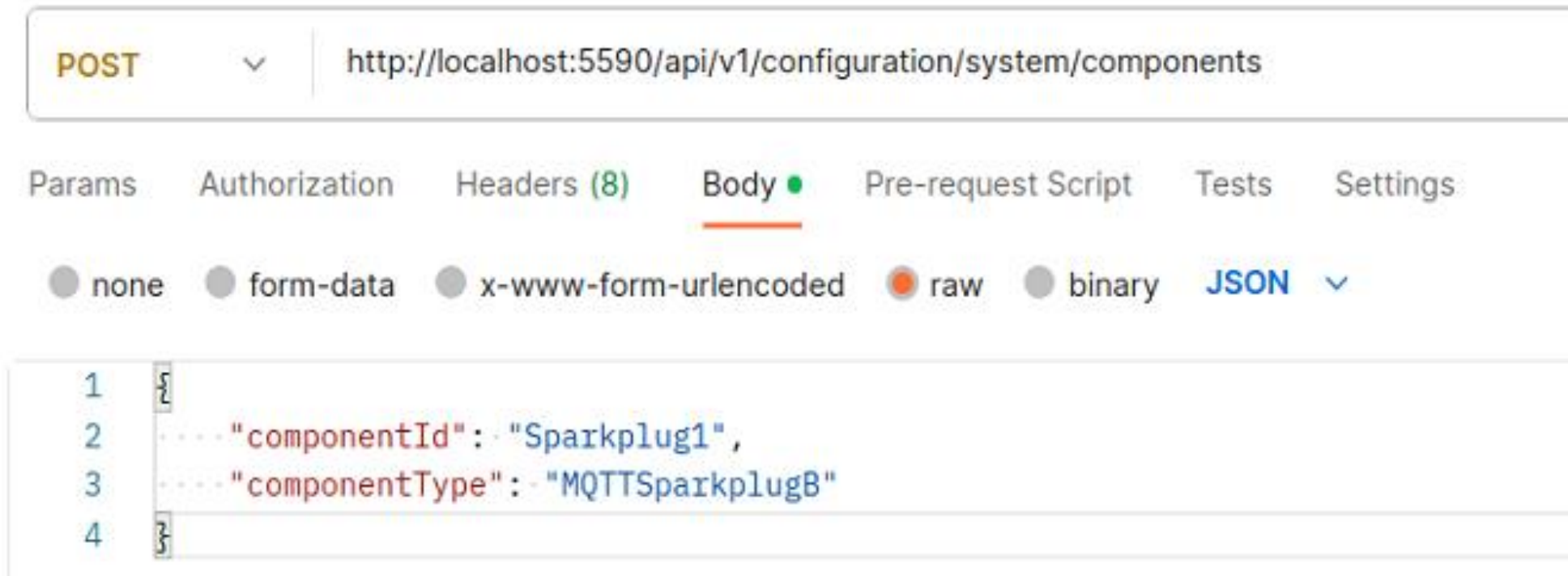
Response ▾



Click Send to get a response

# Configuring the AVEVA Adapter

Create an adapter component – Postman

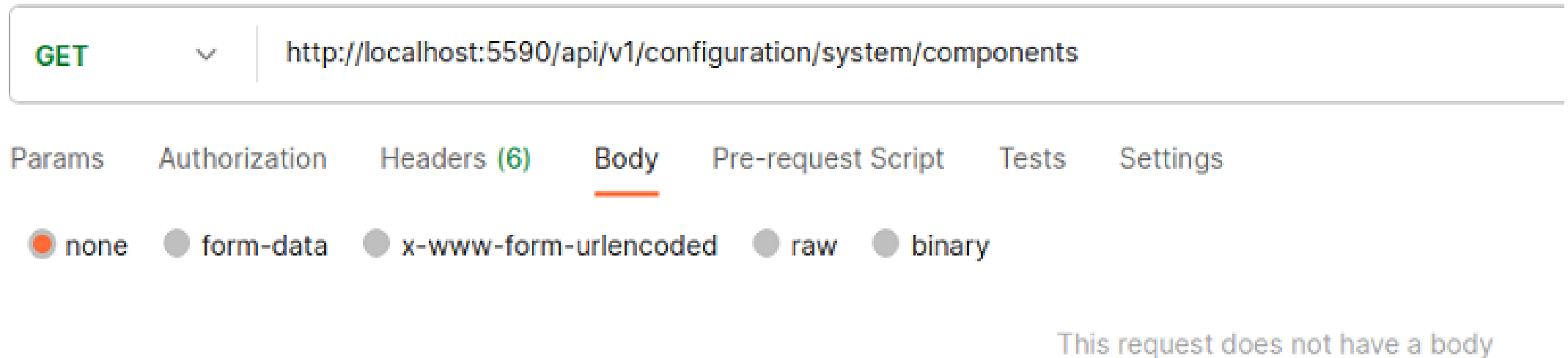


The screenshot shows the Postman interface for a POST request. The URL is `http://localhost:5590/api/v1/configuration/system/components`. The request body is configured as JSON and contains the following content:

```
1 {  
2   ... "componentId": "Sparkplug1",  
3   ... "componentType": "MQTTSparkplugB"  
4 }
```

# Configuring the AVEVA Adapter

Create an adapter component – Postman



# Configuring the AVEVA Adapter

## Configure a data source – Postman

The screenshot shows the Postman interface for configuring a data source. The request method is set to PUT and the URL is `http://localhost:5590/api/v1/configuration/sparkplug1/datasource`. The 'Body' tab is selected, and the content type is set to JSON. The body contains a JSON object with the following fields:

```
1 {  
2   "HostnameOrIpAddress": "10.4.209.213",  
3   "Port": 1883,  
4   "tls": "None",  
5   "username": "adapter",  
6   "password": "hi3"  
7 }
```

PUT http://localhost:5590/api/ + ...

HTTP **http://localhost:5590/api/v1/configuration/sparkplug1/datasource**

📄 Save

</>

PUT ▼ http://localhost:5590/api/v1/configuration/sparkplug1/datasource

Send ▼

Params Authorization Headers (8) **Body** ● Pre-request Script Tests Settings

Cookies

● none ● form-data ● x-www-form-urlencoded ● **raw** ● binary **JSON** ▼

Beautify

```

1  {
2  ... "HostnameOrIpAddress": "10.4.209.213",
3  ... "Port": 1883,
4  ... "tls": "None",
5  ... "username": "adapter",
6  ... "password": "hi3"
7  }

```

Response ▼

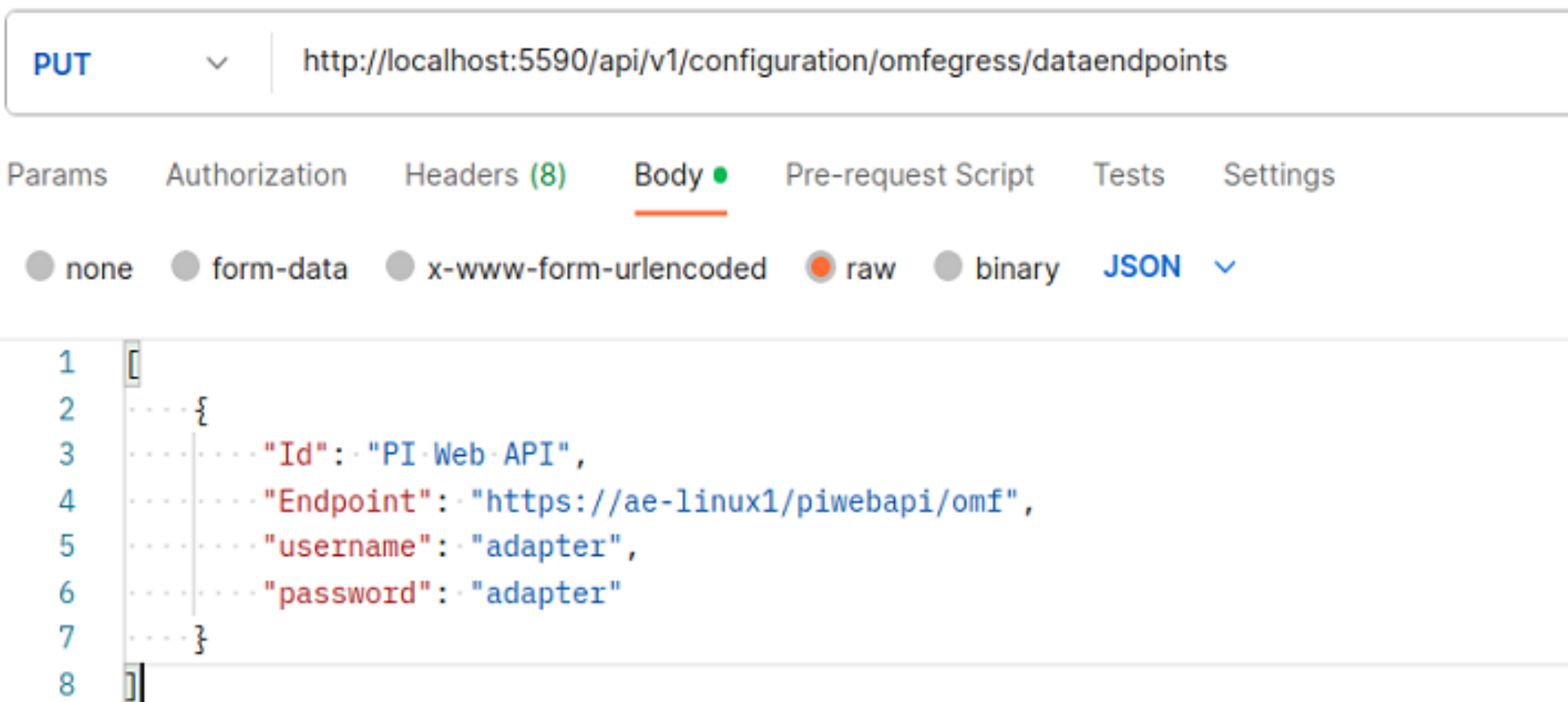


Click Send to get a response



# Configuring the AVEVA Adapter

## Configure a data endpoint – Postman



The screenshot shows the Postman interface for configuring a data endpoint. The request method is set to PUT and the URL is `http://localhost:5590/api/v1/configuration/omfegress/dataendpoints`. The 'Body' tab is selected, and the 'JSON' format is chosen. The request body is a JSON array containing one object with the following properties:

```
1 [
2   {
3     "Id": "PI-Web-API",
4     "Endpoint": "https://ae-linux1/piwebapi/omf",
5     "username": "adapter",
6     "password": "adapter"
7   }
8 ]
```



PUT http://localhost:5590/api/ + ...

 http://localhost:5590/api/v1/configuration/omfegress/dataendpoints

Save



PUT



http://localhost:5590/api/v1/configuration/omfegress/dataendpoints

Send

Params Authorization Headers (8) **Body** Pre-request Script Tests Settings

Cookies

 none  form-data  x-www-form-urlencoded  raw  binary **JSON** ▾

Beautify

```
1 [
2   {
3     "Id": "PI-Web-API",
4     "Endpoint": "https://ae-linux1/piwebapi/omf",
5     "username": "adapter",
6     "password": "adapter"
7   }
8 ]
```

Response



Click Send to get a response

# Configuring the AVEVA Adapter

(Optional) Discover data items – Postman

POST | http://localhost:5590/api/v1/configuration/sparkplug1/discoveries

Params | Authorization | Headers (8) | **Body** | Pre-request Script | Tests | Settings

none  form-data  x-www-form-urlencoded  raw  binary **JSON** ▾

```
1 {  
2   .... "id": "Discovery1"  
3 }
```

POST http://localhost:5590/ap + ⋮

 http://localhost:5590/api/v1/configuration/sparkplug1/discoveries

📄 Save

</>

POST ▾

http://localhost:5590/api/v1/configuration/sparkplug1/discoveries

Send ▾

Params Authorization Headers (8) **Body** ● Pre-request Script Tests Settings

Cookies

Beautify

● none ● form-data ● x-www-form-urlencoded ● raw ● binary **JSON** ▾

```
1 {  
2   ... "id": "Discovery1"  
3 }
```

Response ▾

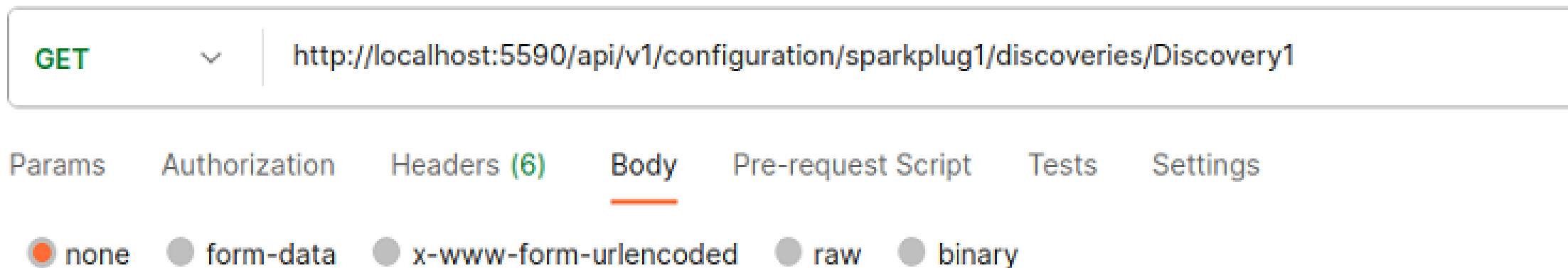


Click Send to get a response

# Configuring the AVEVA Adapter

(Optional) Check discovered data items – Postman

## 1. Get Discovery1 status



The screenshot shows a Postman request configuration for a GET method. The URL is `http://localhost:5590/api/v1/configuration/sparkplug1/discoveries/Discovery1`. The 'Body' tab is selected and underlined, showing radio button options for 'none', 'form-data', 'x-www-form-urlencoded', 'raw', and 'binary'. The 'none' option is selected.

## 2. Get Discovery1 contents



The screenshot shows a Postman request configuration for a GET method. The URL is `http://127.0.0.1:5590/api/v1/Configuration/Sparkplug1/Discoveries/Discovery1/result`.

GET http://localhost:5590/api/ + ...

http://localhost:5590/api/v1/configuration/sparkplug1/discoveries/Discovery1

Save

</>

GET



http://localhost:5590/api/v1/configuration/sparkplug1/discoveries/Discovery1

Send



Params Authorization Headers (6) Body Pre-request Script Tests Settings

Cookies

- none
- form-data
- x-www-form-urlencoded
- raw
- binary

This request does not have a body

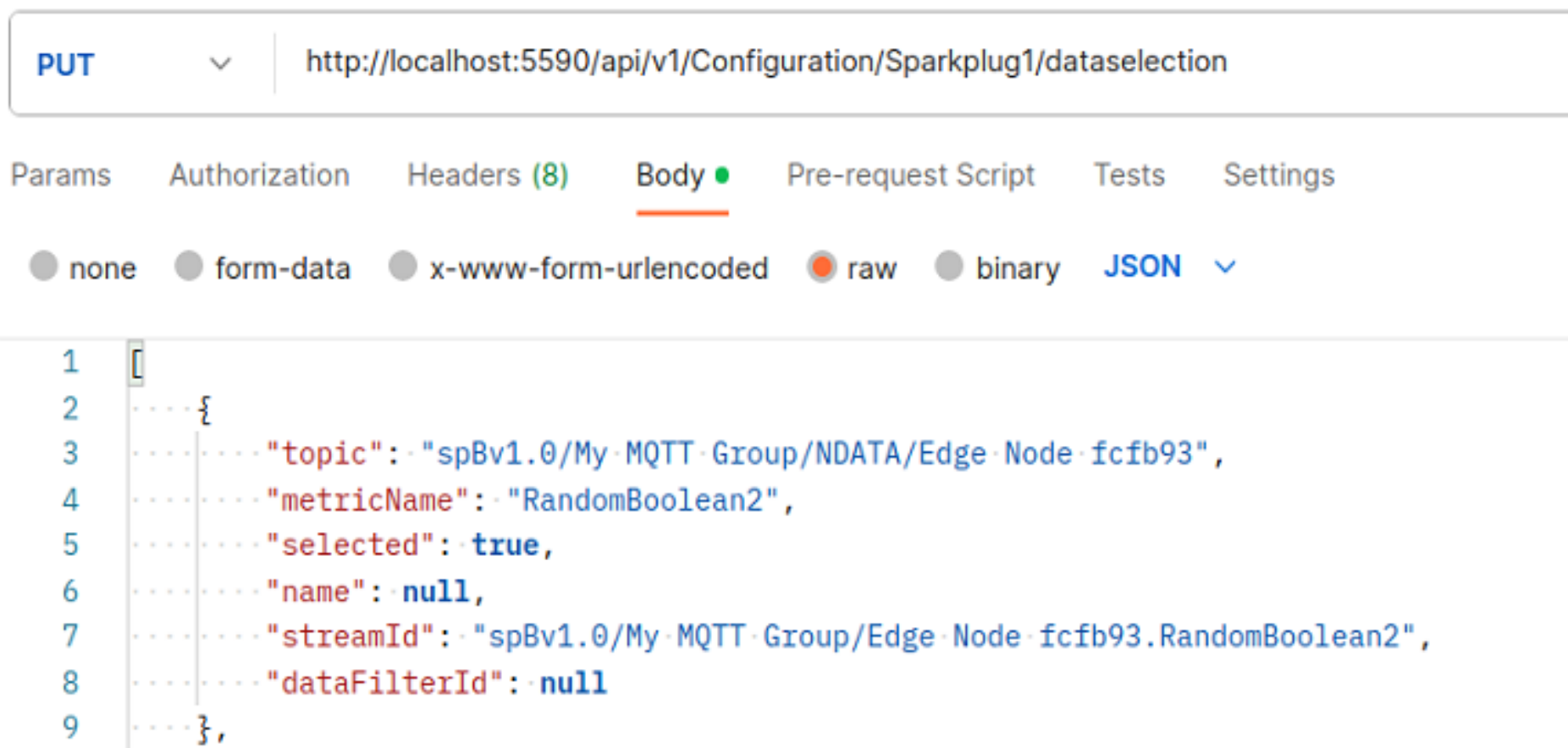
Response



Click Send to get a response

# Configuring the AVEVA Adapter

## Configure data selection – Postman



The screenshot shows the Postman interface for a PUT request. The URL is `http://localhost:5590/api/v1/Configuration/Sparkplug1/dataselection`. The request body is configured as JSON. The body content is a JSON array with one object containing the following fields:

```
1 [
2   {
3     "topic": "spBv1.0/My-MQTT-Group/NDATA/Edge-Node-fcfb93",
4     "metricName": "RandomBoolean2",
5     "selected": true,
6     "name": null,
7     "streamId": "spBv1.0/My-MQTT-Group/Edge-Node-fcfb93.RandomBoolean2",
8     "dataFilterId": null
9   },
10 ]
```



PUT http://localhost:5590/api/ + ⋮



http://localhost:5590/api/v1/Configuration/Sparkplug1/datasetselection

Save



PUT ▾

http://localhost:5590/api/v1/Configuration/Sparkplug1/datasetselection

Send ▾

Params Authorization Headers (8) **Body** ● Pre-request Script Tests Settings

Cookies

 none  form-data  x-www-form-urlencoded  raw  binary **JSON** ▾

Beautify

```
1 [
2   {
3     "topic": "spBv1.0/My MQTT Group/NDATA/Edge Node fcfb93",
4     "metricName": "RandomBoolean2",
5     "selected": true,
6     "name": null,
7     "streamId": "spBv1.0/My MQTT Group/Edge Node fcfb93.RandomBoolean2",
8     "dataFilterId": null
9   },
10  }
```

Response ▾



Click Send to get a response



---

# Recap

- Learned how to install and configure AVEVA Adapter for MQTT and AVEVA Edge Data Store on a Linux environment
  1. Using edgcmd
  2. Using Postman
- Very easy to configure and implement
- Script-ability + access to deploy on many environments

---

# About us



## Ashish Jain

Senior Tech Support Engineer

Escalation team for AVEVA PI Interfaces, Connectors, and Adapters



## Evan Greavu

Senior Tech Support Engineer

Escalation team for AVEVA PI Interfaces, Connectors, and Adapters



# Questions?

Please wait for the microphone.  
State your name and company.



# Please remember to...

Navigate to this session in the mobile app to complete the survey.



# Thank you!

This presentation may include predictions, estimates, intentions, beliefs and other statements that are or may be construed as being forward-looking. While these forward-looking statements represent our current judgment on what the future holds, they are subject to risks and uncertainties that could result in actual outcomes differing materially from those projected in these statements. No statement contained herein constitutes a commitment by AVEVA to perform any particular action or to deliver any particular product or product features. Readers are cautioned not to place undue reliance on these forward-looking statements, which reflect our opinions only as of the date of this presentation.

The Company shall not be obliged to disclose any revision to these forward-looking statements to reflect events or circumstances occurring after the date on which they are made or to reflect the occurrence of future events.

 [linkedin.com/company/aveva](https://www.linkedin.com/company/aveva)

 [@avevagroup](https://twitter.com/avevagroup)

#### ABOUT AVEVA

AVEVA is a world leader in industrial software, providing engineering and operational solutions across multiple industries, including oil and gas, chemical, pharmaceutical, power and utilities, marine, renewables, and food and beverage. Our agnostic and open architecture helps organizations design, build, operate, maintain and optimize the complete lifecycle of complex industrial assets, from production plants and offshore platforms to manufactured consumer goods.

Over 20,000 enterprises in over 100 countries rely on AVEVA to help them deliver life's essentials: safe and reliable energy, food, medicines, infrastructure and more. By connecting people with trusted information and AI-enriched insights, AVEVA enables teams to engineer efficiently and optimize operations, driving growth and sustainability.

Named as one of the world's most innovative companies, AVEVA supports customers with open solutions and the expertise of more than 6,400 employees, 5,000 partners and 5,700 certified developers. The company is headquartered in Cambridge, UK.

Learn more at [www.aveva.com](https://www.aveva.com)